Паттерны проектирования

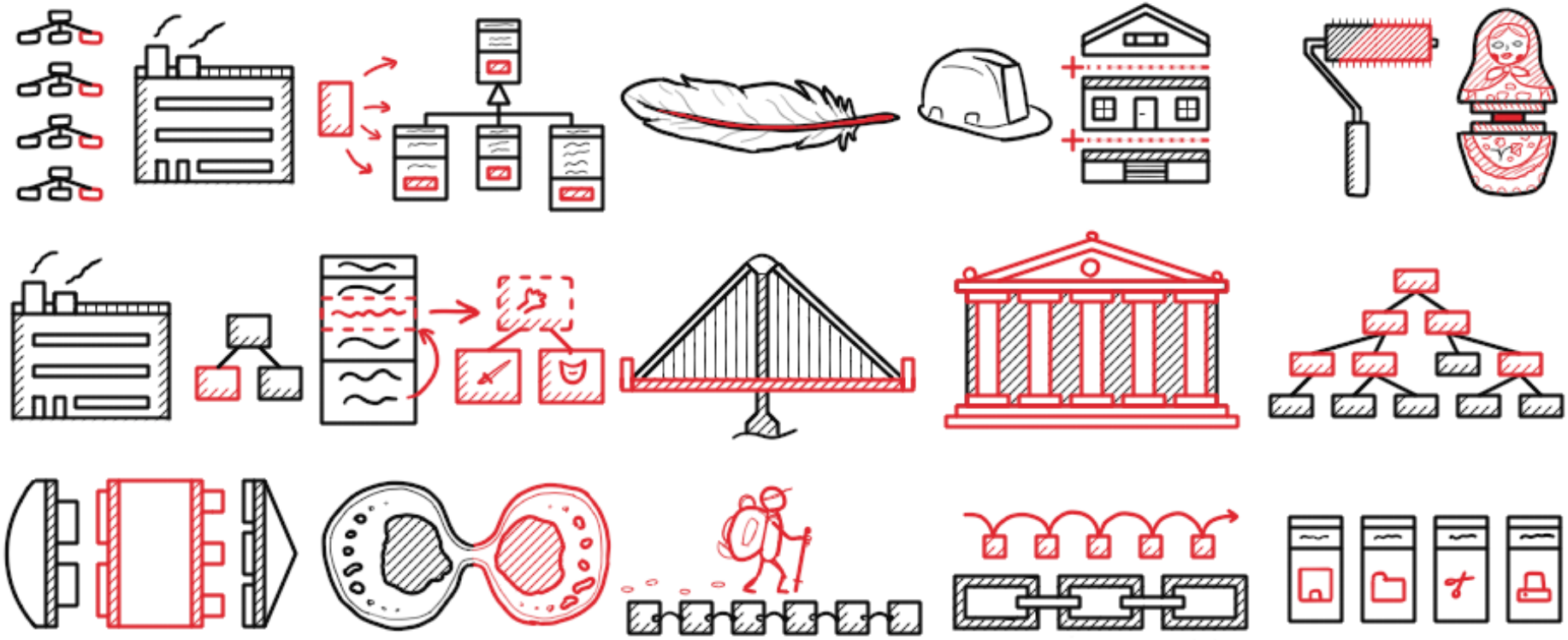
Шаблоны проектирования

Шаблоны проектирования – это стандартные(обобщенные) решения для определенных задач.

- Предложены хорошими специалистами
- Проверены временем
- Повсеместно используемые

В области ПО использование шаблонов проектирования было предложено и развито Gamma, Helms, Johnson и Vlissades – в книге «Design Patterns: Elements of Reusable Object-Oriented Software» 1995

Шаблоны проектирования "банды четырёх (Gang of Four)"



Определение

Под паттерном проектирования понимается описание взаимодействия объектов и классов, адаптированных для решения общей задачи проектирования в конкретном контексте.

Для чего применяются паттерны ?

Для создания “красивого дизайна”

- Инкапсуляция
- Полиморфизм
- Наследование
- Абстракция
- Слабая связанность
- И др.

Классификация паттернов GoF

- Пораждающие (Creational)
- Структурные (Structural)
- Поведенческие (Behavioral)

1. Порождающие шаблоны	2. Структурные шаблоны	3. Шаблоны поведения
1. Abstract Factory	1. Adapter	1. Chain of Responsibility
2. Builder	2. Bridge	2. Command
3. Factory Method	3. Composite	3. Interpreter
4. Prototype	4. Decorator	4. Iterator
5. Singleton	5. Façade	5. Mediator
	6. Flyweight	6. Momento
	7. Proxy	7. Observer
		8. State
		9. Strategy
		10. Template Method
		11. Visitor

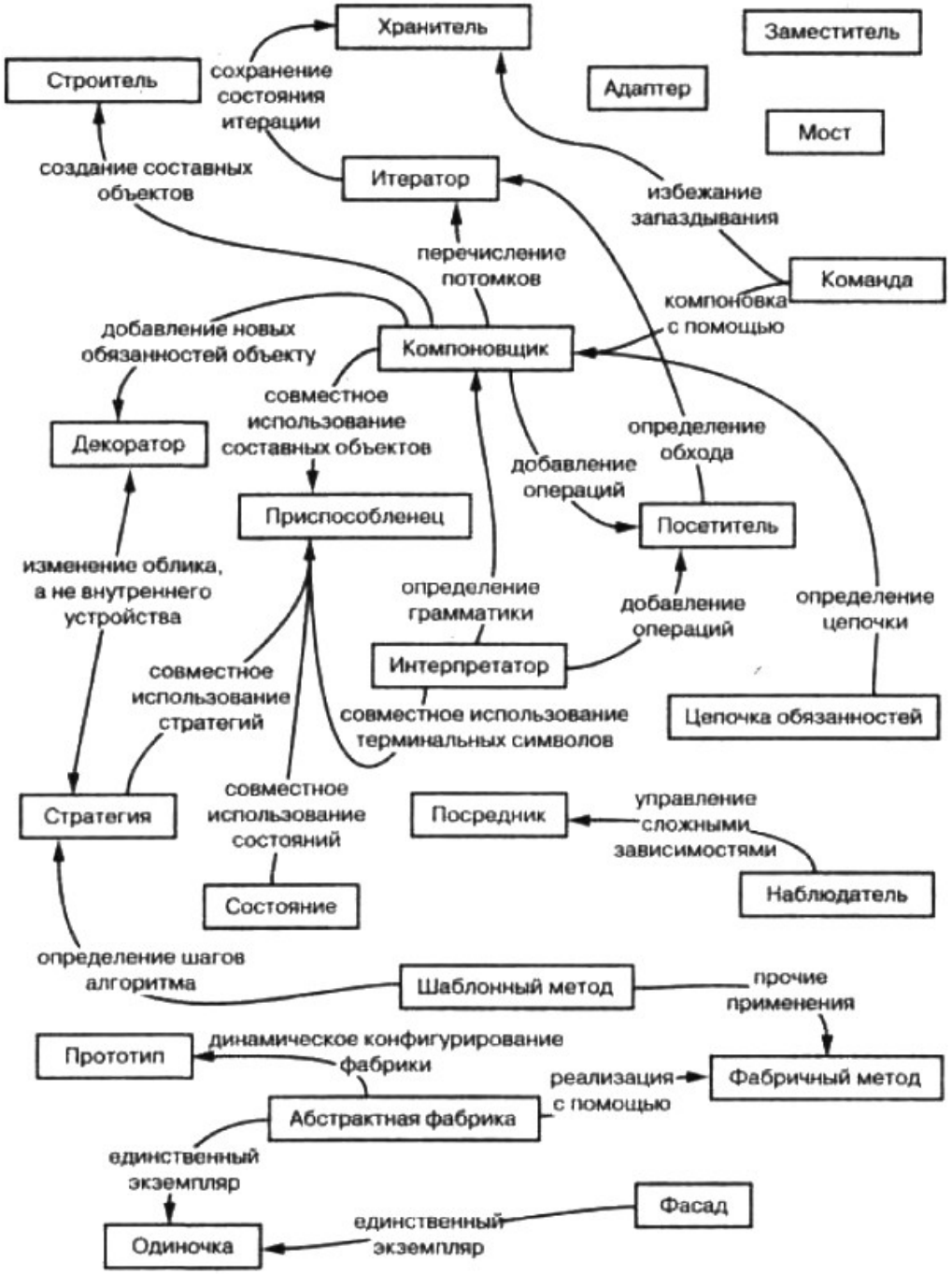


Рис. 1.1. Отношения между паттернами проектирования

Как выбрать паттерн проектирования

- ✔ Подумайте, как паттерны решают проблемы проектирования
- ✔ Пролистайте разделы каталога, описывающие назначение паттернов
- ✔ Изучите взаимосвязи паттернов
- ✔ Проанализируйте паттерны со сходными целями
- ✔ Разберитесь в причинах, вызывающих перепроектирование
- ✔ Посмотрите, что в вашем дизайне должно быть изменяющимся

Когда использовать паттерны ?

- Необоснованное использование паттернов может существенно усложнить программу, поэтому применять паттерны нужно, или когда проблема уже возникает или когда ее появление можно предсказать
- Существует мнение, что применять паттерны надо начинать только тогда, когда программист сам начинает выделять паттерны в своих проектах; до этого момента в большинстве случаев будет иметь место необоснованное их использование
- Но в любом случае познакомиться с ними необходимо хотя бы для того, чтобы как можно раньше появился материал для размышлений

Как пользоваться шаблоном ?

- Прочитайте описание паттерна, чтобы получить о нем общее представление
- Вернитесь назад и изучите разделы «Структура», «Участники» и «Отношения»
- Посмотрите на раздел «Пример кода», где приведен конкретный пример использования паттерна в программе
- Выберите для участников паттерна подходящие имена
  - Обычно имена участников паттерна слишком абстрактны, но иногда бывает удобно включить имена участников паттерна в имена элементов программы (SimpleLayoutStrategy, TeXLayoutStrategy)
- Определите классы
  - Объявите их интерфейсы, установите отношения наследования и определите переменные экземпляра, которыми будут представлены данные объекты и ссылки на другие объекты.
- Определите имена операций, встречающихся в паттерне
  - Будьте последовательны при выборе имен. Например, для обозначения фабричного метода можно было бы всюду использовать префикс Create-.
- Реализуйте операции, которые выполняют обязанности и отвечают за отношения, определенные в паттерне

Графическая нотация OMT (Object Modeling Technique)

- OMT (Object Modeling Technique) – один из методов объектно-ониентированого проектирования и одновременно одна из общепризнанных систем графических обозначений Д. Румбаха (James Rumbaugh)

- Распрастраненной натацией также является UML

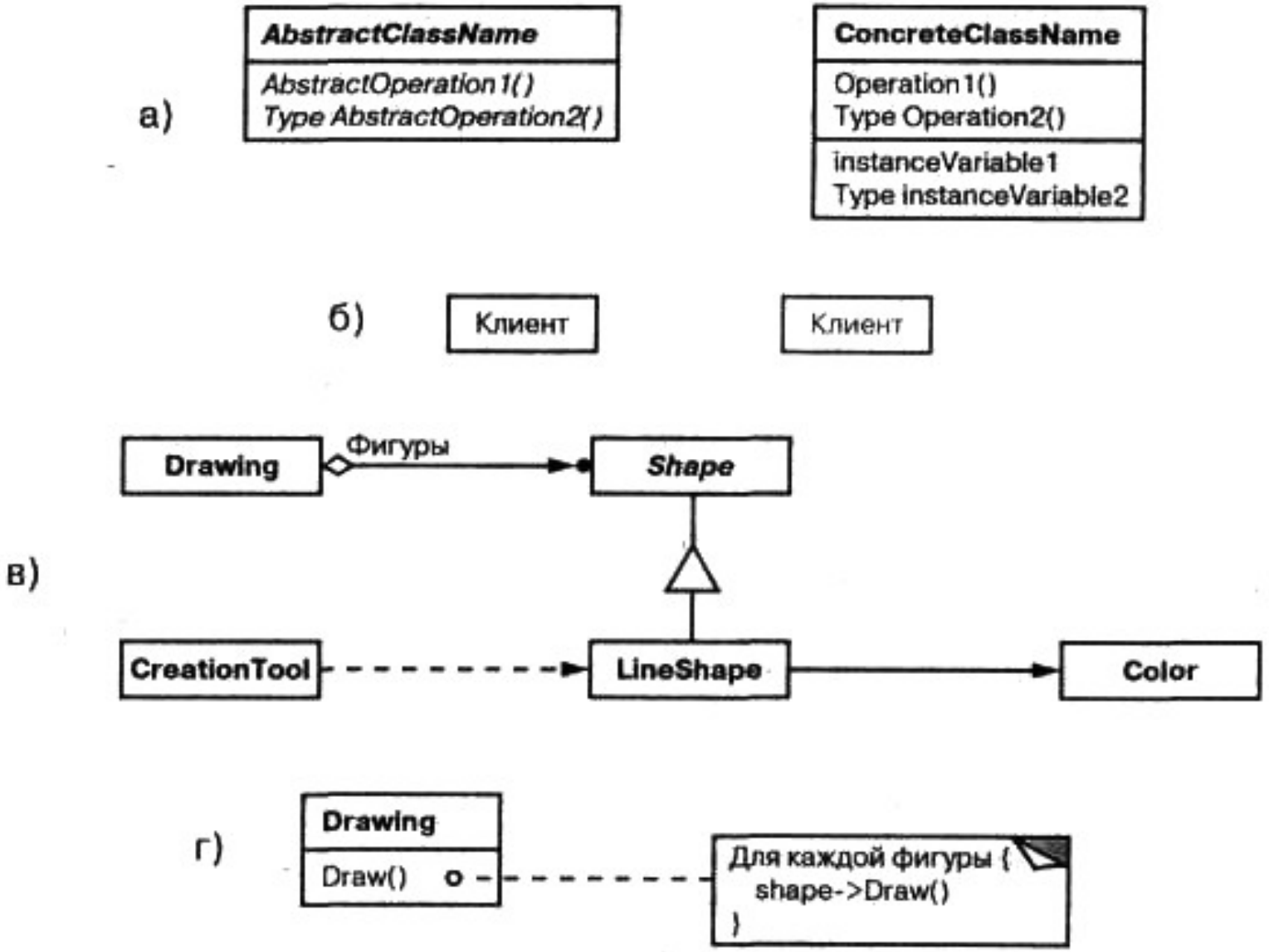


Рис. В.1. Нотация диаграмм классов: а) абстрактные и конкретные класс
б) класс клиента участника (слева) и класс неявного клиента (справа);
в) отношения между классами; г) аннотация на псевдокоде.

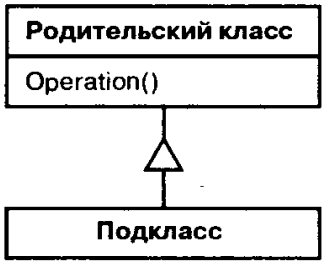
OMT, описание классов

- Класс обозначается прямоугольником
- В верхней части напечатано имя класса
- Описание переменных располагается ниже описания методов
- Можно ставить имя типа перед методом, переменной экземпляра или фактического параметра
- Курсивом в имени обозначаются абстрактные классы (соответственно и интерфейсы) и методы
- При описании паттернов проектирования бледным шрифтом часто обозначают клиентов, которые не входят в состав участников паттерна

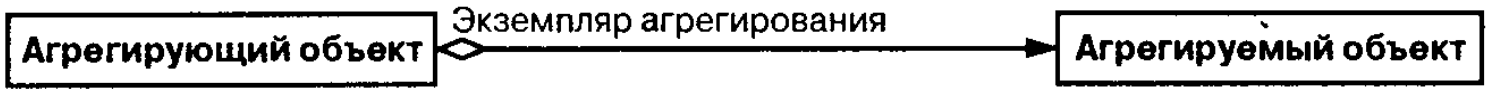
OMT, связи между классами

- Инстанцирование

- Наследование



- Агрегирование

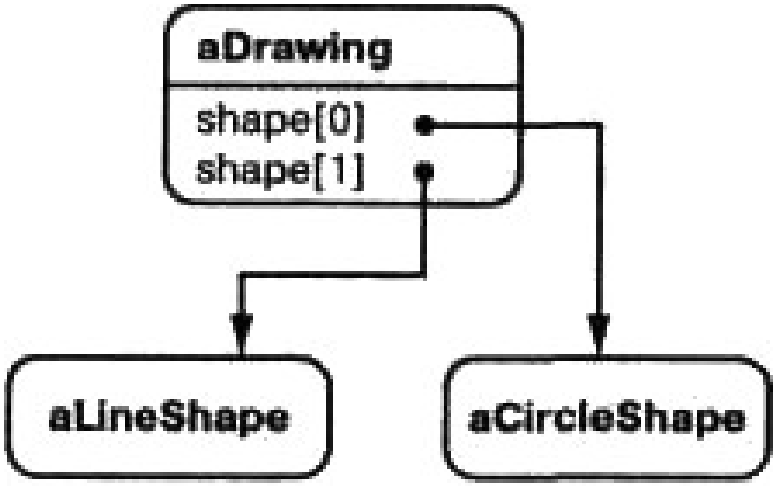


- Осведомленность (ассоциация, использование)

Агрегирование и осведомленность

- И агрегирование и осведомленность – способы композиции объектов и делегирования функциональности
- Агрегирование
  - Один объект владеет другим или несет за него ответственность
  - Агрегат и его составляющие имеют одинаковое время жизни
- Осведомленность
  - Одному объекту известно о другом объекте
  - Осведомленные объекты не несут никакой ответственности друг за друга
- Осведомленность более слабое отношение, чем агрегирование

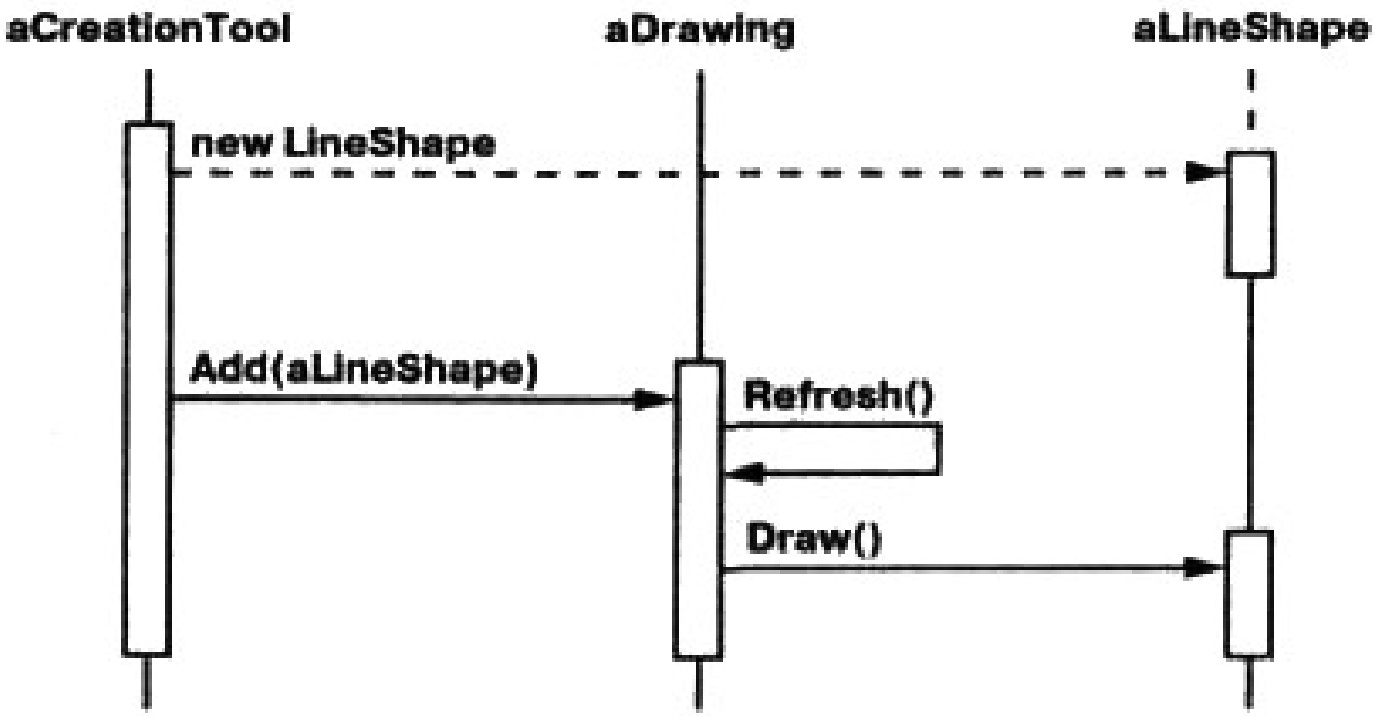
OMT, диаграммы объектов



OMT, описание объектов

- В диаграммах объектов приводятся только экземпляры в какой-то определенный момент времени
- Для обозначения объектов используются прямоугольники со скругленными углами
- Объекты именуются «aSomething», где Something – класс объекта
- Стрелки ведут к объектам, на которые ссылается данный

OMT, диаграммы взаимодействия



OMT, описание взаимодействий

- Время на диаграммах взаимодействий откладывается сверху вниз
- Сплошная вертикальная черта означает время жизни объекта, до момента создания объекта вертикальная линия идет пунктиром
- Вертикальный прямоугольник говорит о том, что объект активен, т.е. обрабатывает какой-либо запрос (выполняет какой-либо метод)
- Соглашение об именовании объектов такое же, как для диаграмм объектов
- Запросы, посылаемые другим объектам (вызовы методов), обозначаются горизонтальной стрелкой, указывающей на объект получатель; имя запроса (метода) показывается над стрелкой
- Запрос на создание объекта обозначается горизонтальной пунктирной стрелкой
- Запрос объекта самому себе изображается стрелкой на этот же объект

Рассмотрим фабричный метод (Factory Method),

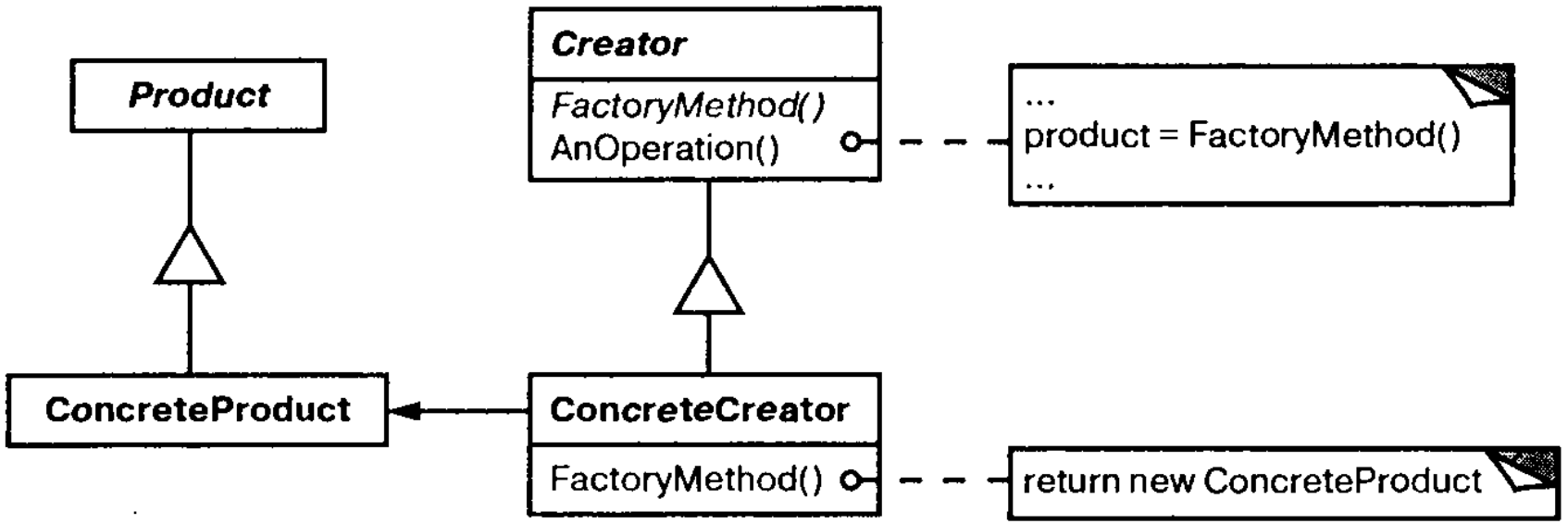
Задача

- Определяет интерфейс для создания объекта абстрактного (редко конкретного) типа, но скрывает способ создания

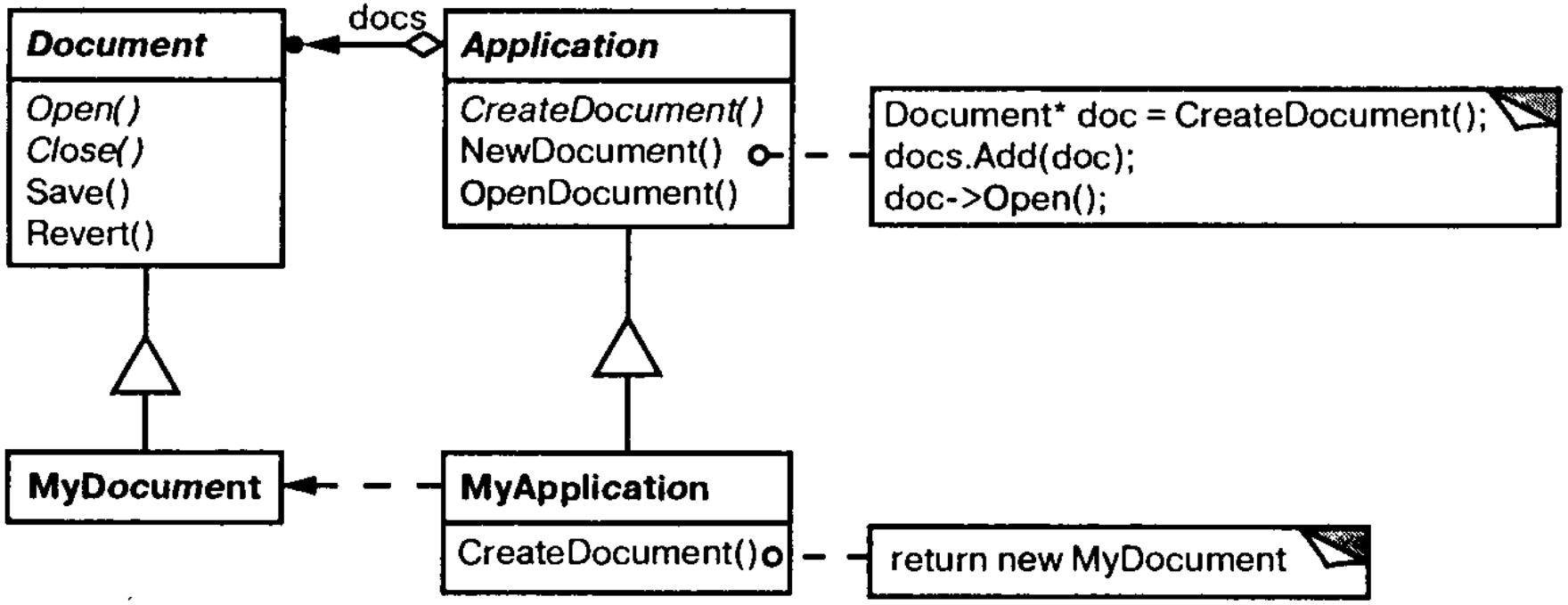
- Фабричный метод позволяет классу делегировать инстанцирование подклассам

Фабричный метод (Factory Method), диаграмма классов

- Делегирование инстанцирования подклассам
  - В данном конкретном случае является частным случаем абстрактной фабрики с одним видом продукта



Фабричный метод (Factory Method), пример с созданием документов



Перейдем к реализации фабричного метода на практике.